Area Optimization on Fixed Analog Floorplans using Convex Area Functions

A. Unutulmaz
School of Electrical and Electronics Engineering, Bogazici University
Email: unutulm@boun.edu.tr

G. Dündar
School of Electrical and Electronics Engineering, Bogazici University
Email: dundar@boun.edu.tr

F.V. Fernández
Institute of Microelectronics of Sevilla, CSIC and University of Sevilla
Email: pacov@imse-cnm.csic.es

***Abstract*—A methodology to optimize the area of a fixed non-slicing floorplan is presented in this paper. Areas of transistors, capacitors and resistors are formulated as convex functions and area is minimized by solving a sequence of convex problems. The methodology is practical even with many components and variants. Moreover symmetry constraints are satisfied during optimization.**

I. Introduction

Reducing iterations between different design levels has been suggested by the International Technology Roadmap for Semiconductors as a major contribution towards the reduction of design cost. Towards this end, the integration of physical and electrical synthesis in one single step has been proposed, yielding the so-called layout-aware circuit synthesis approaches, such as [1]. Most successful circuit synthesis approaches are based on the formulation of the sizing problem as an optimization problem, commonly solved by iterative processes that imply hundreds or thousands of circuit performance evaluations. It becomes obvious that integration of physical synthesis into such circuit synthesis process is only practical if the circuit layout can be instanced very fast. From the existing layout synthesis approaches, layout templates meet such speed constraint. The quality of a layout instance depends on parameters such as matching, aspect ratio constraints and area utilization. In this paper, a methodology to optimize area utilization of a given floorplan is presented. Widths and heights of analog components in the layout are realistically modelled and areas of these components are formulated as convex functions. This approach may be easily integrated with the Layout Description Script [2], which is based on Linear Programming, capable of handling placement as well as routing, and suitable for automatic layout template synthesis. Thus, the presented approach, combined with LDS, will allow area optimization on an LDS template in a layout in the loop approach.

Roughly speaking, layout quality is proportional to area occupation, which fundamentally depends on placement of its composing cells and the appropriate selection of the variant of each cell. If the floorplan of a layout is fixed, optimization is reduced only to variant selection. In general, a floorplan is classified either slicing or non-slicing. If a floorplan is representable by a polish expression [3], it is called a slicing floorplan; else, non-slicing. The term *non-slicing floorplan* does not represent the complement set of slicing floorplans, it covers any type of floorplan.

978-3-9815370-0-0/DATE13/

Area optimization on fixed floorplans has been studied for more than three decades. Shape curves (functions) were used to solve the area minimization (area optimization) problem on a slicing floorplan in [4]. The procedure is as follows. Shape functions for all components are constructed. A shape function is a set containing $width$ and $height$ pairs. All sets are combined hierarchically, keeping only the best solutions. The method efficiently finds the optimal sizes for the components; however, not only the slicing structure limits the solution space, but also symmetry constraints cannot be directly satisfied within the shape function approach. Symmetric devices must have same dimensions; if dimension of one is changed, the other must be simultaneously changed. Shape function for a component is constructed by changing geometry parameters of the component and then calling the relevant device generator. For instance, a geometrical parameter for a transistor is its finger count m. Transistors are generated for different numbers of fingers, e.g. number fingers m is varied from 1 to 20. Note that at every call to the device generator, only one variant is generated.

It may be pointed out that by a sequence of compactions a non-slicing floorplan may be obtained from a slicing one [5], thus working with slicing floorplans does not effect the solution. However, it must be pointed out that in this case area will be optimized for a slicing floorplan not for a non-slicing one.

Geometric programming was applied in area optimization on non-slicing floorplans in [6]. Efficiency of the approach was improved in [7] by reducing the number of variables and constraints. However, geometric programming formulation cannot handle symmetry constraints which is a must for analog floorplans. In [8] area optimization was performed by solving a sequence of linear programs iteratively and linear approximations were used to approximate shape functions. For soft, variable size, blocks these approaches assumed constant area which is not the case for analog layouts. As an example,

TABLE I
COMPARISON OF THE METHODS IN [4], [6]–[9] AND THIS WORK

Method	Solution Type	Soft Blocks	Analog Constraints	Floorplan Type	Optimization Method
[4]	Global	Shape Functions	Not Considered	Slicing	Hierarchical
[6]	Global	Constant Area	Not Considered	Non-Slicing	Geometric Programming
[7]	Global	Constant Area	Not Considered	Non-Slicing	Similar to [6]
[8]	Local	Constant Area (Approximated)	Not Considered	Non-Slicing	Linear Programming
[9]	Global	Enhanced Shape Functions	Handled?	Non-Slicing	Hierarchical, Enumeration
[This Paper]	Global	Area Functions	Handled	Non-Slicing	Convex Programming

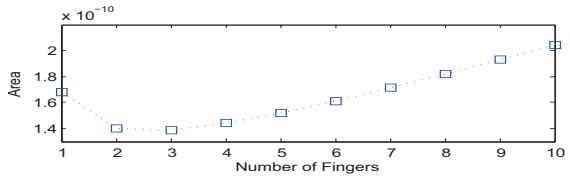


Fig. 1. Area of a transistor, with fixed W and L, is not constant as the number of fingers change.

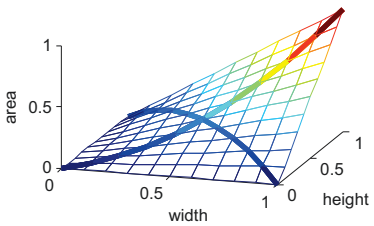


Fig. 2. Area is not a convex function of $(width, height)$. Convex and concave cuts are indicated as tick curves.

Fig. 1 shows the area of a transistor for different numbers of fingers.

Area minimization on a non-slicing floorplan was also studied in [9]. Variants are combined in an enhanced shape function by enumerating all possible solutions. Enumeration is costly if the number of variants is high. Also, it is not clear how to construct the enhanced shape functions for two sets of modules having additional super constraints of symmetry.

In Table I, the methodology of this work is compared with the methodologies of the works in [4], [6]–[9].

Contributions of this paper may be summarized as follows:

1) Realistic layout models for transistor, capacitor and resistor layouts are presented and areas for these models are shown to be convex.
2) Under integer relaxation, the optimum dimensions of the layout components are found by solving a sequence of convex problems.
3) During the optimization, symmetry constraints are satisfied. Dimensions of symmetric modules are equated. These equalities are linear constraints and are added to the optimization problem.

The rest of the paper is organized as follow: A brief discussion about convex functions is given in Section II. Section III presents *area functions* and discusses convexity of an area function. In Section IV, a sequential optimization methodology is presented. Results for test circuits are given in Section V. Finally, Section VI concludes the paper.

II. CONVEX FUNCTIONS AND OPTIMIZATION

A function ($f : \mathbf{R}^n \rightarrow \mathbf{R}$) is convex, if f satisfies

$$f(\theta * x + (1-\theta) * y) \leq \theta * f(x) + (1-\theta) * f(y) \qquad (1)$$

for all $x, y \in \mathbf{R}^n$ and for all $\theta \in \mathbf{R} : 0 \leq \theta \leq 1$. Thus, a local minimum of a convex function f is in fact its global minimum.

Solution of a convex problem may be easily obtained using commercial solvers. The MOSEK solver [10] has been used in this work. This solver supports quadratic (2) and rotated quadratic (3) cone constraints in addition to linear optimization problems. Conic constraints must have one of these forms:

$$x_1 \geq \sqrt{\sum_{j=2}^{n} x_j^2} \qquad (2)$$

$$2 * x_1 * x_2 \geq \sqrt{\sum_{j=3}^{n} x_j^2} \qquad (3)$$

where $x_i \in \mathbf{R}$, n is an integer ($n > 2$ for (2) and $n > 3$ for (3)) and in (3) x_1 and x_2 must be non-negative.

Detailed information can be found in [11].

III. AREA FUNCTIONS AND CONVEXITY

The problem of area optimization is not a convex problem. In Fig. 2,the area of a rectangular module as function of width and height is plotted. In this figure, convex and concave cuts are shown to indicate the non-convexity of the problem. Non-convexity prevents convex formulation of the problem. The problem may be converted to a geometric program as in [6], but then alignment and symmetric placement constraints can not be satisfied.

In this paper, sequential convex programming is applied to solve the problem of area optimization. Although the problem is not convex, the $width/height$ Pareto-Front appears to be convex when areas of layout components are formulated as convex functions. In Section III-A, *Area Functions* for transistors, capacitors and resistors are presented and in Section III-B, convexity of the total area of the layout is investigated.

A. Area Functions

An area function returns the area of a component given its geometrical parameters. For instance, number of fingers,

metal width, W, L are some of the geometrical parameters of a transistor. Resistors and capacitors also have similar geometrical parameters. Some of these parameters depend on the technology process and are fixed for a device generator. Some of them are *input parameters* such as W and L of a transistor, and the rest are *free parameters* and affect the geometry of the layout such as the number of fingers of a resistor.

We present functions for transistors, capacitors and resistors. Although parameter values in these functions are specific for a device generator. Parameters may be easily extracted by investigating a few instances of a device generator. We extracted the parameters for AMS $0.35\mu m$ CMOS technology. Given the template, device generators, and input parameters, area optimization is performed by varying the free parameters.

1) Transistors: A model for a single transistor is shown in Fig. 3a. Formulas for the height and width are given in (4) and (5).

$$height(m) = \frac{W}{m} + \alpha_1 \tag{4}$$

$$width(m) = m * (L + \alpha_2) + \alpha_3 \tag{5}$$

where m is the number of fingers, W is the channel width, L is the channel length, α_is are constants. For a given device generator, the parameters (α_i) are fixed.

The transistors in Fig. 3a and Fig. 3b are synthesized by different device generators; thus, the values of the parameter α_i are different. However the formulation in (4) and (5) is general enough to cover both device generators. In other words, α parameters contain all the required information about orientation, guard-ring, dummies, routing, etc...

Given the parameters α_i, W and L, area of the transistor will be a function of the number of fingers m. Area is simply $height * width$ and equals to:

$$area(m) = \left(\frac{W}{m} + \alpha_1\right) * (m * (L + \alpha_2) + \alpha_3) \tag{6}$$

By eliminating m, the $width$ in (5) may be rewritten as a function of $height$ as in (7), where for a valid transistor ($m \geq 1$), $height$ is always greater than α_1. Then the area function given in (6) may be reformulated as in (8).

$$width(height) = \frac{W * (L + \alpha_2)}{height - \alpha_1} + \alpha_3 \tag{7}$$

$$area(height) = height * \left(\frac{W * (L + \alpha_2)}{height - \alpha_1} + \alpha_3\right) \tag{8}$$

The condition in (1) boils down to the expression in (9) when f is replaced by the area function in (8). For all θ satisfying $0 \leq \theta \leq 1$ and for all x and y greater than α_1, the expression in (9) is true. Thus, the area function in (8) satisfies the convexity condition in (1) and it is a convex function. Similarly, convexity may be shown by writing (8) as a function of $width$.

$$0 \leq \frac{W * \alpha_1 * \theta * (L + \alpha_2) * (x - y)^2 * (1 - \theta)}{((x - \alpha_1) * (y - \alpha_1) * (y * (1 - \theta) + \theta * x - \alpha_1))} \tag{9}$$

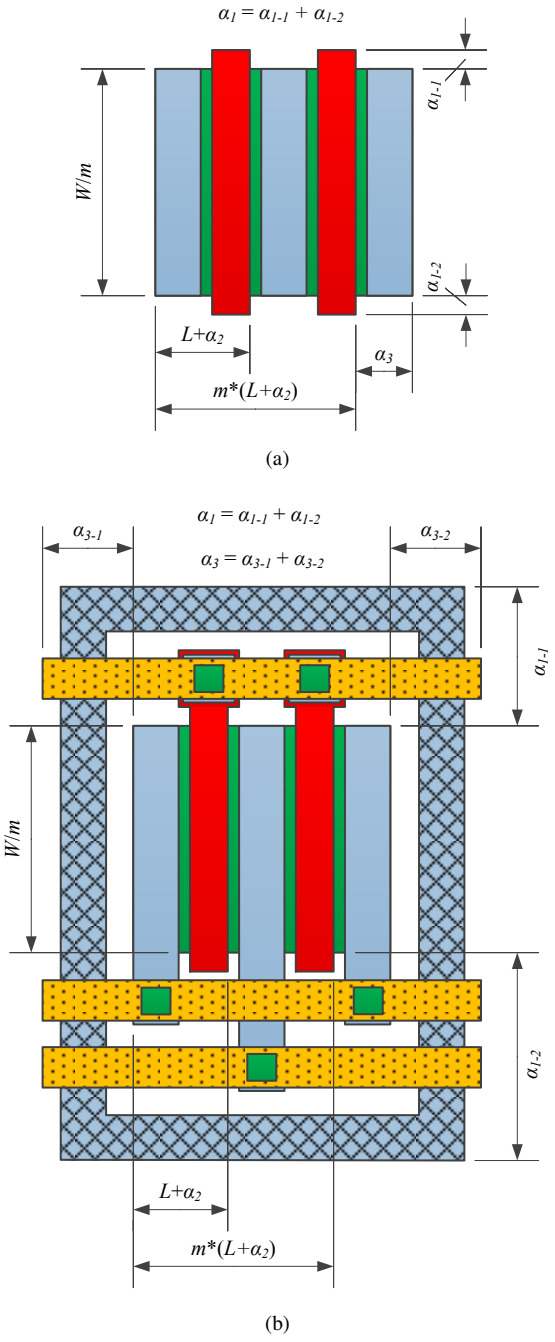


Fig. 3. Transistor parameters are shown on two different transistors: (a) Simple Transistor, (b) Transistor with Guard-ring and Routing

2) Capacitors: Dimensions for capacitors are modeled via the formulas in (10) and (11):

$$height(w) = \frac{C}{C_x * w} + \beta_1 \tag{10}$$

$$width(w) = w + \beta_2 \tag{11}$$

where C is total capacitance, C_x and β_is are parameters of the device generator, w is the width of the overlap region between the capacitor plates. A capacitor and its parameters are shown

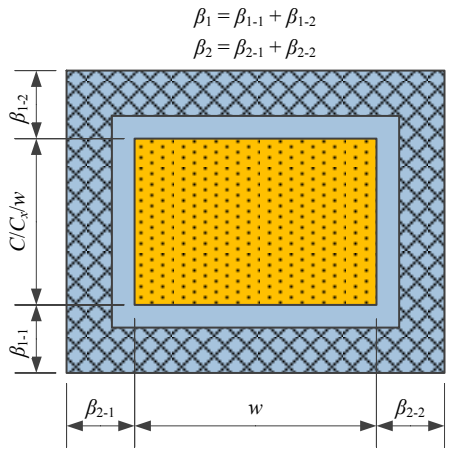


Fig. 4. Parameters of a Capacitor

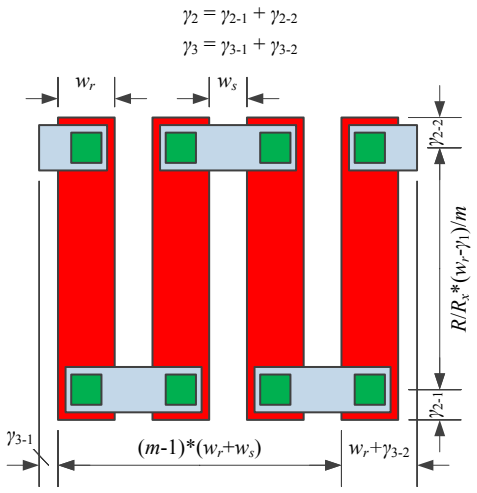


Fig. 5. Parameters of a Resistor

in Fig. 4. The area function of a capacitor is:

$$area = height * width = \left(\frac{C}{C_x * w} + \beta_1\right) * (w + \beta_2) \quad (12)$$

and it may be shown to be a convex function.

3) Resistors: The dimensions of a resistor are shown in Fig. 5 and modeled by the following formulas:

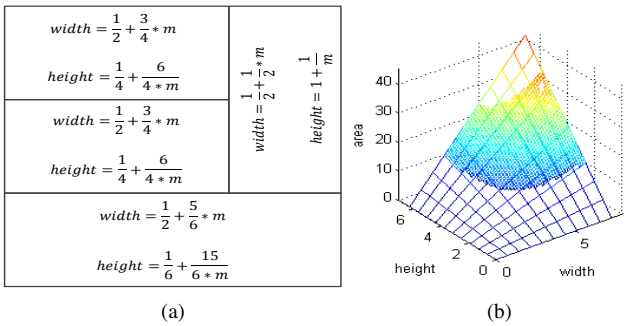
$$height(m) = \frac{\frac{R}{R_x} * (w_r - \gamma_1)}{m} + \gamma_2 \quad (13)$$

$$width(m) = (m-1) * (w_r + w_s) + w_r + \gamma_3 \quad (14)$$

where R is the resistance. Parameters w_r , w_s, R_x and γ_i are the device generator parameters and m is the number of fingers.

The area function of a resistor is given in (15) and it may also be proven to be a convex function.

$$area = \left(\frac{\frac{R}{R_x} * (w_r - \gamma_1)}{m} + \gamma_2\right) * ((m-1) * (w_r + w_s) + w_r + \gamma_3) \quad (15)$$



(a) (b)

Fig. 6. (a) A Placement and Functions for Dimensions, (b) Area Plot: obtained by varying all ms in (a) from 1 to 5, Pareto-Front ($width/height$) in (b) is observed to be convex.

B. Convexity of the Layout Area

Although the presented area functions are convex, convexity of the total layout area must be investigated. As an introduction, a floorplan of four components and their dimension models are shown in Fig. 6a. For this layout, the number of fingers m for all components are the free parameters. Keeping the floorplan fixed and sweeping all m's from 1 to 5, the area plot in Fig. 6b is obtained. In this plot, the Pareto-Front ($width/height$) may be observed to be a convex function of $(width, height)$. Lemma 1 and the following discussion states the convexity.

Lemma 1: If all the components in a layout have convex area functions and there is no dead space in the layout, the total area is going to be convex.

Proof: If there is no dead space, then the total area of the layout is going to be the sum of the component areas. If there are n blocks, the total area is going to be:

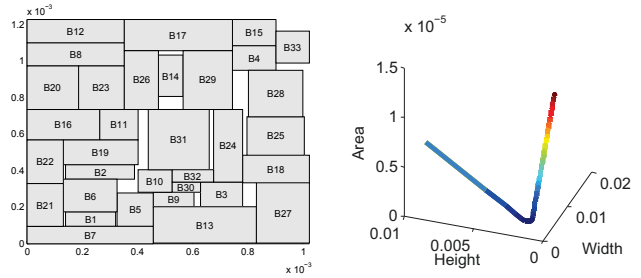
$$area_{total} = \sum_{i=1}^{n} area_i \quad (16)$$

The sum in (16) is convex, due to the fact that non-negative weighted sum of convex function is convex [11]. ■

In case when the layout has dead space, the Pareto-Front appeared to be convex. We tested the convexity using the *ami33* circuit from the *MCNC* benchmarks. We have generated hundreds of random placements and investigated the convexity. Component areas in the benchmark are kept constant and the aspect ratios are allowed to be free in the interval $[2/3, 3/2]$. Pareto-Fronts are plotted and all of them are observed to be convex. A placement and the corresponding Pareto-Front is shown in Fig. 7.

IV. Area Optimization

Although layout area appears to be a convex function on the Pareto-Front, the objective function $height * width$, alone, is not a convex function. Thus the area minimization problem can not be formulated in MOSEK [10]. Fortunately, it is possible to optimize the problem by solving a sequence of convex problems. The procedure is as follows:



(a) (b)

Fig. 7. (a) A Placement of *ami33* from *MCNC* Benchmark, (b) Pareto Frontier for the Placement in (a)

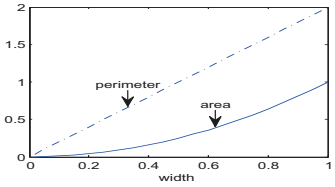


Fig. 8. Changes in $perimeter$ and $area$ are plotted, where $width$ is swept from 0 to 1 and $height \cong width$. If perimeter is minimized, area is also minimized.

1) For the first iteration, the perimeter is used as the objective function which is convex.
2) Iteratively, solutions for width and height are used to weigh the objective function.
3) When the solution converges, the optimization is terminated.

A. Approximation for Area

Due to the fact that the non-convex objective function $area = height * width$ can not be formulated in a convex program, the objective function is approximated by the affine function

$$objective = \lambda * \left(\frac{width}{\tilde{r}} + height \right) \tag{17}$$

where $\tilde{r}$ is an approximation for the aspect ratio r, $\lambda \in \mathbf{R}$ is a constant and may be chosen as 1.

The gradient for $area = height * width$ is:

$$\begin{aligned} \nabla area &= \left(\frac{\partial area}{\partial width}, \frac{\partial area}{\partial height} \right) \\ &= (height, width) \cong (height, \tilde{r} * height) \end{aligned} \tag{18}$$

Similarly, the gradient for the objective function in (17) is:

$$\begin{aligned} \nabla objective &= \left(\frac{\partial objective}{\partial width}, \frac{\partial objective}{\partial height} \right) \\ &= \left(\frac{\lambda}{\tilde{r}}, \lambda \right) \end{aligned} \tag{19}$$

Although the magnitudes of the two gradients in (18) and (19) are different, their directions are close to each other. If aspect ratio is close to $\tilde{r}$, minimizing (17) will also minimize the area.

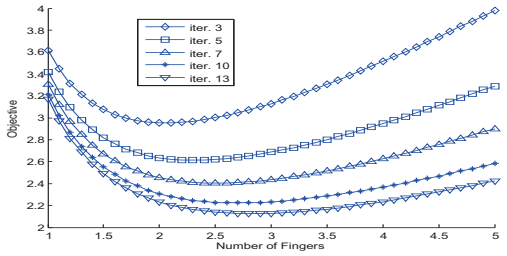


Fig. 9. Change in Objective Function

For example, if r is known to be close to unity, then (17) can be written as

$$objective = 2 * (height + width) \tag{20}$$

which is simply the perimeter. Minimizing the perimeter will also minimize area which can be seen from the plot in Fig. 8.

B. Methodology

Unfortunately, we do not know the exact aspect ratio before optimization so we can not use the objective function in (17). Therefore, we applied an iterative approach. First, the aspect ratio r in (17) is initialized to unity ($r_0 = 1$) and the problem is optimized. Resulting dimensions ($width_1$ and $height_1$) are used to calculate the aspect ratio ($r_1 = width_1/height_1$) for the next iteration. Next iterations use the solution of the previous ones. When the algorithm converges, the resulting dimensions are the optimal ones. The objective function may be formulated as:

$$objective_n = \frac{width}{r_{n-1}} + height \tag{21}$$

where r_{n-1} is:

$$r_{n-1} = \frac{width_{n-1}}{height_{n-1}} \tag{22}$$

For the bottom component in Fig. 6a, changes in the objective function, according to (21), are plotted in Fig. 9. For this component, optimal area is obtained when $m = 3$ and this may be also be observed from Fig. 9, where the minimum of the objective function shifts to 3.

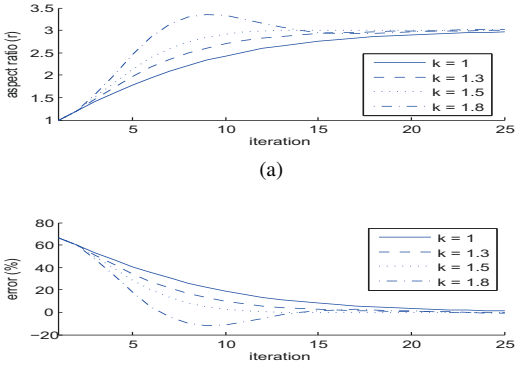
Lastly, the objective in (21) is modified as (23) to improve the convergence of the method. Detailed discussion is given in Section IV-D.

$$objective_n = \frac{width}{r_{n-2} + k * (r_{n-1} - r_{n-2})} + height \tag{23}$$

where $k \in \mathbf{R}$ and $1 \leq k$. Objective in (23) boils down to (21) for $k = 1$.

C. Solving the Iterations

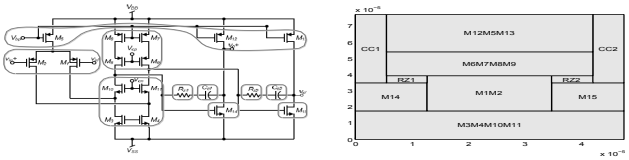
Iterations are solved using the MOSEK [10] optimizer. Below, the formulation of a transistor for MOSEK is given.



(a)

(b)

Fig. 10. (a) Convergence of r to $r_{opt} = 3$, (b) Percentage Error



(a) (b)

Fig. 11. Fully Differential Amplifier

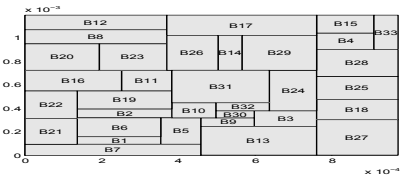


Fig. 12. A Placement for *ami33* Circuit

The $height$ in (4) is formulated via an inequality and rotated cone constraints:

$$height \geq x_{rq} + \alpha_1 \tag{24}$$

$$2 * x_{rq} * m \geq \sqrt{x_c^2} \tag{25}$$

$$x_c = 2 * W \tag{26}$$

where α_1 and W are constants; $height$, m are variables and x_c and x_{rq} are intermediate variables. The $width$ in (5) is formulated as an equality constraint:

$$width = (m * (L + \alpha_2) + \alpha_3) \tag{27}$$

where α_2, α_3 and L are constants; $width$ and m are variables. Capacitors and resistors are reformulated for MOSEK, analogously.

D. Convergence Analysis

Convergence of the method is tested for several examples. The number of iterations mainly depends on the optimal aspect ratio r_{opt}; the closer r_{opt} to unity, the less number of iterations. For the bottom component in Fig. 6a, $r_{opt} = 3$, convergence of the method and percentage error are plotted in Fig. 10 for different k values in (23). By experience, $k = 1.3$ is a good choice for k. Convergence is observed to be fast initially; however, slows down when the error is small. Our algorithm stops when change in the dimensions $height_n - height_{n-1}$ and $width_n - width_{n-1}$ are smaller then $100nm$.

V. Results

We tested the methodology on a fully differential amplifier in Fig. 11a and on *ami33* circuit from *MCNC* benchmarks. For the differential amplifier, area functions were extracted from device generators implemented on AMS $0.35\mu m$ CMOS technology. Resulting layout is shown in Fig. 11b. The optimization terminated after 8 iterations and optimization took $20msec$. The component areas for *ami33* benchmark circuit are kept constant at their original values in *MCNC* benchmarks. Aspect ratio r for all components are left free in the range $[1/3 \leq r \leq 3]$ and the floorplan in Fig. 7a is used. Optimization terminated after 2 iterations and took $13msec$. All the tests were done on an Intel i7-3610QM processor with $6GByte$ RAM.

VI. Conclusion

This paper presents layout models for transistors, capacitors and resistors, where areas of these models are shown to be convex functions. Also a discussion about the convexity of the layout area is presented. Under integer relaxation, optimum dimensions of the layout components are found by solving a sequence of convex problems. During the optimization also analog constraints, such as symmetry and alignment are satisfied. The methodology is tested on a fully differential amplifier and on a benchmark floorplan. Computation times are promising and we are improving the methodology to handle integer constraints, e.g., number of fingers must be an integer number. This methodology may be easily integrated with the template description language in [2].

References

[1] R. Castro-Lopez, O. Guerra, E. Roca and F.V. Fernandez, *"An integrated layout-synthesis approach for analog ICs,"* IEEE Trans. on Computer-Aided Design, vol. 27, no. 7, pp. 1179-1189, July 2008.

[2] A. Unutulmaz, G. Dundar and F.V. Fernandez, *"LDS - A Description Script for Layout Templates,"* Proc. European Conf. on Circuit Theory and Design, pp. 857-860, 2011.

[3] D. F. Wong and C. L. Liu, *"A new algorithm for floorplan design,"* Proc. of the 23rd ACM/IEEE Design Automation Conf., pp. 101-107, 1986.

[4] R. Otten, *"Efficient Floorplan Optimization,"* Proc. of the International Conf. on Computer Design, pp. 499-502, 1983.

[5] M. Lai and D. Wong. 2001. *"Slicing tree is a complete floorplan representation,"* Proc. of the Design, automation and test in Europe Conf., pp. 228-232, 2001.

[6] T.-S. Moh, T.-S. Chang, and S. L. Hakimi, *"Globally optimal floorplanning for a layout problem,"* IEEE Trans. On Circuit and Systems I, vol. 43, pp. 713-720, Sep. 1996.

[7] H. Murata and E. S. Kuh, *"Sequence-pair based placement method for hard/soft/preplaced modules,"* Proc. Int. Symp. Physical Design, pp. 167-172, 1998.

[8] Chen P, Kuh ES, *"Floorplan sizing by linear programming approximation,"* Proc. of ACM/IEEE Design Automation Conf., pp. 468-471, 2000.

[9] Martin Strasser, et al. *"Deterministic Analog Circuit Placement using Hierarchically Bounded Enumeration and Enhanced Shape Functions,"* Proc. of International Conf. on Computer-Aided Design, pg. 306-313, 2008.

[10] *www.mosek.com/*

[11] S. Boyd and L. Vandenberghe, *"Convex Optimization,"* Cambridge University Press, 2009.